MOUNT

```
MM      MM    AAAAAA    KK      KK  LL              000000      GGGGGGG
MM      MM    AAAAAA    KK      KK  LL              000000      GGGGGGGG
MMMM  MMMM   AA    AA   KK      KK  LL            00    00  GG
MMMM  MMMM   AA    AA   KK      KK  LL            00    00  GG
MM  MM  MM   AA    AA   KK    KK    LL            00    00  GG
MM  MM  MM   AA    AA   KK    KK    LL            00    00  GG
MM      MM   AA    AA   KKKKK       LL            00    00  GG
MM      MM   AA    AA   KKKKK       LL            00    00  GG    GGGGGG
MM      MM   AAAAAAAAAA KK    KK    LL            00    00  GG    GGGGGG
MM      MM   AAAAAAAAAA KK    KK    LL            00    00  GG        GG   ....
MM      MM   AA    AA   KK      KK  LL            00    00  GG        GG   ....
MM      MM   AA    AA   KK      KK  LLLLLLLLL    000000      GGGGGG        ....
MM      MM   AA    AA   KK      KK  LLLLLLLLL    000000      GGGGGG        ....

LL              IIIIII       SSSSSSSS
LL              IIIIII       SSSSSSSS
LL                II       SS
LL                II       SS
LL                II       SS
LL                II       SS
LL                II         SSSSSS
LL                II         SSSSSS
LL                II             SS
LL                II             SS
LL                II             SS
LL                II             SS
LLLLLLLLL       IIIIII     SSSSSSSS
LLLLLLLLL       IIIIII     SSSSSSSS
```

MAKLOG

M 9
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 1
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (1)

```
    1    0001  0 MODULE MAKLOG (
    2    0002  0               LANGUAGE (BLISS32),
    3    0003  0               IDENT = 'V04-000'
    4    0004  0               ) =
    5    0005  1 BEGIN
    6    0006  1
    7    0007  1 !
    8    0008  1 !****************************************************************
    9    0009  1 !*                                                              *
   10    0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                     *
   11    0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.      *
   12    0012  1 !*  ALL RIGHTS RESERVED.                                        *
   13    0013  1 !*                                                              *
   14    0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
   15    0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE *
   16    0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER *
   17    0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
   18    0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY *
   19    0019  1 !*  TRANSFERRED.                                                *
   20    0020  1 !*                                                              *
   21    0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE *
   22    0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT *
   23    0023  1 !*  CORPORATION.                                                *
   24    0024  1 !*                                                              *
   25    0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS *
   26    0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.     *
   27    0027  1 !*                                                              *
   28    0028  1 !*                                                              *
   29    0029  1 !****************************************************************
   30    0030  1
   31    0031  1 !++
   32    0032  1 !
   33    0033  1 ! FACILITY:  MOUNT Utility Structure Levels 1 & 2
   34    0034  1 !
   35    0035  1 ! ABSTRACT:
   36    0036  1 !
   37    0037  1 !     These routines allocate and hook up the logical name and mounted
   38    0038  1 !     volume list entries.
   39    0039  1 !
   40    0040  1 ! ENVIRONMENT:
   41    0041  1 !
   42    0042  1 !     STARLET operating system, including privileged system services
   43    0043  1 !     and internal exec routines.
   44    0044  1 !
   45    0045  1 !--
   46    0046  1 !
   47    0047  1 !
   48    0048  1 ! AUTHOR: Andrew C. Goldstein,  CREATION DATE: 20-Oct-1977  19:30
   49    0049  1 !
   50    0050  1 ! MODIFIED BY:
   51    0051  1 !
   52    0052  1 !     V03-018 HH0041         Hai Huang              24-Jul-1984
   53    0053  1 !             Remove REQUIRE 'LIBD$:[VMSLIB.OBJ]MOUNTMSG.B32'.
   54    0054  1 !
   55    0055  1 !     V03-017 HH0040         Hai Huang              20-Jul-1984
   56    0056  1 !             Cal EXE$CRE_GTABLE to create the logical name table if
   57    0057  1 !             it doesn't already exist.
```

MAKLOG
V04-000

N 9
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 2
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1 (1)

```
 58      0058  1 !
 59      0059  1 !   V03-016 HH0038          Hai Huang              12-Jul-1984
 60      0060  1 !           Correct MOUNT_FLAGS structure attribute.
 61      0061  1 !
 62      0062  1 !   V03-015 MHB0153         Mark Bramhall          27-Apr-1984
 63      0063  1 !           Correct NSA$B_ARG_FLAG setting for multiple audits enabled.
 64      0064  1 !
 65      0065  1 !   V03-014 ACG0423         Andrew C. Goldstein,   24-Apr-1984  11:06
 66      0066  1 !           Make disk logical names in MOUNT exec mode to make them
 67      0067  1 !           available to privileged programs.
 68      0068  1 !
 69      0069  1 !   V03-013 HH0012          Hai Huang              09-Apr-1984
 70      0070  1 !           Get the device owner UIC and the volume protection
 71      0071  1 !           from the ORB instead of the UCB.
 72      0072  1 !
 73      0073  1 !   V03-012 HH0009          Hai Huang              27-Mar-1984
 74      0074  1 !           Add security auditing support.
 75      0075  1 !
 76      0076  1 !   V03-011 HH0007          Hai Huang              21-Mar-1984
 77      0077  1 !           Add cluster-wide group-volume support, i.e., create the
 78      0078  1 !           group logical name in the group table of the current
 79      0079  1 !           process.
 80      0080  1 !
 81      0081  1 !   V03-010 HH0002          Hai Huang              23-Jan-1984
 82      0082  1 !           Add job-wide mount support.
 83      0083  1 !
 84      0084  1 !   V03-009 ROW0254         Ralph O. Weber         12-NOV-1983
 85      0085  1 !           Cause logical names defined herein to be of the GETDVI
 86      0086  1 !           FULLDEVNAM form.  This will result in allocation class names
 87      0087  1 !           being used for the equivalence name strings of logical names
 88      0088  1 !           defined by mount.  Therefore, the mounted volume logical name
 89      0089  1 !           equivalence strings can be held over time and passed around
 90      0090  1 !           the VAXcluster without becomming stale.
 91      0091  1 !
 92      0092  1 !   V03-008 CDS0001         Christian D. Saether   2-Aug-1983
 93      0093  1 !           Remove references to RVX structure (obselete).
 94      0094  1 !
 95      0095  1 !   V03-007 DMW4057         DMWalp                 23-Jun-1983
 96      0096  1 !           Change $xxLNM value parameters to be by reference
 97      0097  1 !
 98      0098  1 !   V03-006 DMW4050         DMWalp                 15-Jun-1983
 99      0099  1 !           Corrections to DMW4033; added LNM$M_TERMINAL
100      0100  1 !           Change over to LNM$_LNMB_ADDR
101      0101  1 !
102      0102  1 !   V03-005 ADE9004         A.ELDRIDGE             29-May-1983
103      0103  1 !           Fixed name binding to logical name tables.
104      0104  1 !
105      0105  1 !   V03-004 DMW4033         DMWalp                 26-May-1983
106      0106  1 !           Intergate new logical name structures.
107      0107  1 !
108      0108  1 !   V03-003 STJ50311        Steven T. Jeffreys,    10-Feb-1982
109      0109  1 !           - Make all uses of PHYS_NAME indexed by DEVICE_INDEX,
110      0110  1 !             but always use PHYS_NAME[0] for tape mounts.
111      0111  1 !           - Set the access mode of the logical names(s) created
112      0112  1 !             to be the MIN (PSL$C_SUPER,.CALLERS_ACMOD).  (SPR 45688)
113      0113  1 !
114      0114  1 !   V03-002 DMW4010         DMWalp                 19-Nov-1982
```

MAKLOG
V04-000

B 10
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 3
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (1)

```
; 115        0115  1 !                  Rework logical name block to MTL ( or UCB ) links.
; 116        0116  1 !
; 117        0117  1 !      V03-001 STJ0248          Steven T. Jeffreys,     31-Mar-1982
; 118        0118  1 !      - Allow for ASCII "A" characters in a volume name.
; 119        0119  1 !
; 120        0120  1 !      V02-006 STJ0205          Steven T. Jeffreys,     07-Feb-1982
; 121        0121  1 !      Create a local copy of the user specified logical name
; 122        0122  1 !      to prevent it from being stepped on.
; 123        0123  1 !
; 124        0124  1 !      V02-005 LMP0006          L. Mark Pilant,         29-Dec-1981 12:00
; 125        0125  1 !      Interlock the mount list to avoid potential disasters.
; 126        0126  1 !
; 127        0127  1 !      V02-004 ACG0219          Andrew C. Goldstein,    23-Oct-1981  10:48
; 128        0128  1 !      Add concealed device support in MOUNT
; 129        0129  1 !
; 130        0130  1 !      V02-003 STJ0122          Steven T. Jeffreys,     10-Sep-1981
; 131        0131  1 !      Fixed references to the logical name descriptor to use
; 132        0132  1 !      the symbolic offsets.  This ensures that references to
; 133        0133  1 !      the logical name length will be WORD context.
; 134        0134  1 !
; 135        0135  1 !      V02-002 ACG0167          Andrew C. Goldstein,    18-Apr-1980  13:38
; 136        0136  1 !      Previous revision history moved to MOUNT.REV
; 137        0137  1 !**
; 138        0138  1
; 139        0139  1
; 140        0140  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
; 141        0141  1 REQUIRE 'SRC$:MOUDEF.B32';
; 142        0673  1
; 143        0674  1
; 144        0675  1 LITERAL
; 145        0676  1          PHYS_LENGTH      = 15;                   ! longest allowable physical name
; 146        0677  1
; 147        0678  1 FORWARD ROUTINE
; 148        0679  1          LABEL_LENGTH;                           ! return the length of a volume label
```

MAKLOG
V04-000

C 10
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page   4
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1 (2)

```
 150   0680  1 GLOBAL ROUTINE ALLOC_LOGNAME (MODE) : NOVALUE =
 151   0681  1
 152   0682  1 !++
 153   0683  1 !
 154   0684  1 ! FUNCTIONAL DESCRIPTION:
 155   0685  1 !
 156   0686  1 !     This routine allocates the mounted volume list entry from the
 157   0687  1 !     appropriate storage pools.  It used to allocate logical name block
 158   0688  1 !     also ( thus the name ).
 159   0689  1 !
 160   0690  1 !
 161   0691  1 ! CALLING SEQUENCE:
 162   0692  1 !     ALLOC_LOGNAME ()
 163   0693  1 !
 164   0694  1 ! INPUT PARAMETERS:
 165   0695  1 !     MODE: 0 to use user-specified logical name
 166   0696  1 !           1 to force use of volume name
 167   0697  1 !
 168   0698  1 ! IMPLICIT INPUTS:
 169   0699  1 !     MOUNT parser database
 170   0700  1 !
 171   0701  1 ! OUTPUT PARAMETERS:
 172   0702  1 !     NONE
 173   0703  1 !
 174   0704  1 ! IMPLICIT OUTPUTS:
 175   0705  1 !     MTL_ENTRY: address of MTL block
 176   0706  1 !
 177   0707  1 ! ROUTINE VALUE:
 178   0708  1 !     NONE
 179   0709  1 !
 180   0710  1 ! SIDE EFFECTS:
 181   0711  1 !     NONE
 182   0712  1 !
 183   0713  1 !--
 184   0714  1
 185   0715  2 BEGIN
 186   0716  2
 187   0717  2 EXTERNAL
 188   0718  2     MOUNT_OPTIONS    : BITVECTOR,    ! command options
 189   0719  2     MTL_ENTRY        : REF BBLOCK;   ! MTL block
 190   0720  2
 191   0721  2 EXTERNAL ROUTINE
 192   0722  2     ALLOCATE_MEM;                    ! allocate dynamic memory
 193   0723  2
 194   0724  2
 195   0725  2 ! Now allocate the mounted volume list entry.
 196   0726  2 ! Note: to support job-wide mount, a mount list entry
 197   0727  2 ! is always allocated from paged pool.
 198   0728  2 !
 199   0729  2 MTL_ENTRY = ALLOCATE_MEM (MTL$C_LENGTH, 1);
 200   0730  2
 201   0731  2 MTL_ENTRY[MTL$B_TYPE] = DYN$C_MTL;
 202   0732  2
 203   0733  1 END;                                 ! end of routine ALLOC_LOGNAME

                                                .TITLE  MAKLOG
```

MAKLOG
V04-000
```
                              D 10
                              16-Sep-1984 01:16:19    VAX-11 Bliss-32 V4.0-742        Page  5
                              14-Sep-1984 12:45:22    DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (2)
```

```
                                                .IDENT  \V04-000\

                                                .EXTRN  MOUNT_OPTIONS, MTL_ENTRY
                                                .EXTRN  ALLOCATE_MEM

                                                .PSECT  $CODE$,NOWRT,2

                          0000 00000             .ENTRY  ALLOC_LOGNAME, Save nothing       ; 0680
                       01  DD 00002             PUSHL   #1                                 ; 0729
                       18  DD 00004             PUSHL   #24
              0000G CF 02  FB 00006             CALLS   #2, ALLOCATE_MEM
              0000G CF 50  D0 0000B             MOVL    R0, MTL_ENTRY
              0A  A0 19  90 00010             MOVB    #25, 10(R0)                          ; 0731
                          04 00014             RET                                        ; 0733
```

; Routine Size: 21 bytes,   Routine Base: $CODE$ + 0000

```
  205    0734  1   GLOBAL ROUTINE ENTER_LOGNAME (UCB, VCB) : NOVALUE =
  206    0735  1
  207    0736  1   !++
  208    0737  1   !
  209    0738  1   !   FUNCTIONAL DESCRIPTION:
  210    0739  1   !
  211    0740  1   !       This routine completes the logical name and mounted volume list
  212    0741  1   !       entries.  It builds MTL entry and creates the logical name
  213    0742  1   !       and hooks up the MTL entry in the appropriate list.
  214    0743  1   !
  215    0744  1   !
  216    0745  1   !   CALLING SEQUENCE:
  217    0746  1   !       ENTER_LOGNAME (ARG1, ARG2)
  218    0747  1   !
  219    0748  1   !   INPUT PARAMETERS:
  220    0749  1   !       ARG1: UCB of volume being mounted
  221    0750  1   !       ARG2: VCB of volume being mounted
  222    0751  1   !
  223    0752  1   !   IMPLICIT INPUTS:
  224    0753  1   !       MOUNT parser data base
  225    0754  1   !       MTL_ENTRY: address of MTL block
  226    0755  1   !       SMTL_ENTRY: address of MTL block for volume set
  227    0756  1   !
  228    0757  1   !   OUTPUT PARAMETERS:
  229    0758  1   !       NONE
  230    0759  1   !
  231    0760  1   !   IMPLICIT OUTPUTS:
  232    0761  1   !       NONE
  233    0762  1   !
  234    0763  1   !   ROUTINE VALUE:
  235    0764  1   !       NONE
  236    0765  1   !
  237    0766  1   !   SIDE EFFECTS:
  238    0767  1   !       logical name and MTL entry entered
  239    0768  1   !
  240    0769  1   !--
  241    0770  1
  242    0771  2   BEGIN
  243    0772  2
  244    0773  2   MAP
  245    0774  2           UCB                 : REF BBLOCK,    ! UCB being mounted
  246    0775  2           VCB                 : REF BBLOCK;    ! VCB being mounted
  247    0776  2
  248    0777  2   BUILTIN
  249    0778  2           INSQUE,
  250    0779  2           CALLG;
  251    0780  2
  252    0781  2   BIND
  253    0782  2           TAPE_PREFIX         = UPLIT BYTE ( 'TAPE$' ),
  254    0783  2           DISK_PREFIX         = UPLIT BYTE ( 'DISK$' ),
  255    0784  2           SYSTEM_TABLE        = %ASCID 'LNM$SYSTEM',
  256    0785  2           JOB_TABLE           = %ASCID 'LNM$JOB';
  257    0786  2
  258    0787  2   LOCAL
  259    0788  2           ACMODE,                                 ! access mode
  260    0789  2           INDEX,                                  ! local index into PHYS_NAME vector
  261    0790  2           P,                                      ! string pointer
```

```
262  0791  2              C,                               ! string count
263  0792  2              RVT             : REF BBLOCK,    ! pointer to RVT
264  0793  2              NAME_DESC       : BBLOCK [DSC$K_S_BLN],
265  0794  2                                               ! internal logical name descriptor
266  0795  2              LOG_BUFFER      : VECTOR [LNM$C_NAMLENGTH,BYTE],
267  0796  2                                               ! logical name buffer
268  0797  2              MOUNT_LIST      : REF BBLOCK,     ! address of mount list tail
269  0798  2              ITEM_LIST       : VECTOR [(6*3)+1,LONG],
270  0799  2                                               ! $CRELNM item list, 6 items each
271  0800  2                                               ! 3 longwords in lenght plus 1
272  0801  2                                               ! for the terminator longword
273  0802  2              PHYSNAM_DESC    : BBLOCK [ DSC$K_S_BLN ],
274  0803  2                                               ! GETDVI descriptor for physical name
275  0804  2              FULLNAM         : VECTOR [ PHYS_LENGTH + 2, BYTE ],
276  0805  2                                               ! Place to store the FULLDEVNAM string
277  0806  2              DVI_ITEM        : VECTOR [ 3+1, LONG ],
278  0807  2                                               ! GETDVI item list
279  0808  2              JIB             : REF BBLOCK,     ! pointer to Job Info Block
280  0809  2              TABLE_NAME      : VECTOR [16, BYTE]
281  0810  2                                INITIAL (%ASCII 'LNM$GROUP_000000'),
282  0811  2                                               ! Group table name
283  0812  2              GROUP_TABLE     : VECTOR [2, LONG]
284  0813  2                                INITIAL (16, TABLE_NAME),
285  0814  2                                               ! Group table name descriptor
286  0815  2              ASC_GROUP       : VECTOR [8, BYTE]
287  0816  2                                INITIAL (%ASCII '00000000')
288  0817  2                                               ! Group in ASCII (6 bytes used)
289  0818  2              ASC_GROUP_DESC  : VECTOR [2, LONG]
290  0819  2                                INITIAL (6, ASC_GROUP);
291  0820  2                                               ! ASCII group descriptor
292  0821  2
293  0822  2  EXTERNAL
294  0823  2              MOUNT_OPTIONS   : BITVECTOR,      ! command options
295  0824  2              MOUNT_FLAGS     : VECTOR,         ! mount flags
296  0825  2              CALLERS_ACMOD   : LONG,           ! Caller's (of $MOUNT) access mode
297  0826  2              DEVICE_CHAR     : BBLOCK,         ! device characteristics
298  0827  2              DEVICE_COUNT,                    ! number of devices specified
299  0828  2              LOG_NAME        : VECTOR,         ! logical name descriptor
300  0829  2              DEVICE_INDEX    : LONG,           ! index into PHYS_NAME vector
301  0830  2              PHYS_NAME       : VECTOR,         ! physical device name descriptor
302  0831  2              MTL_ENTRY       : REF BBLOCK,     ! MTL block
303  0832  2              SMTL_ENTRY      : REF BBLOCK,     ! MTL block for volume set
304  0833  2              SCH$GL_CURPCB   : REF BBLOCK ADDRESSING_MODE (GENERAL),
305  0834  2                                               ! address of our PCB
306  0835  2              IOC$GQ_MOUNTLST : VECTOR ADDRESSING_MODE (GENERAL),
307  0836  2                                               ! system mounted volume list head
308  0837  2              EXE$GL_FLAGS    : BITVECTOR ADDRESSING_MODE (GENERAL),
309  0838  2                                               ! exec flags longword
310  0839  2              NSA$GR_ALARMVEC : BBLOCK ADDRESSING_MODE (GENERAL),
311  0840  2                                               ! alarm enable bit vector
312  0841  2              NSA$GR_JOURNVEC : BBLOCK ADDRESSING_MODE (GENERAL);
313  0842  2                                               ! journal enable bit vector
314  0843  2
315  0844  2  EXTERNAL LITERAL
316  0845  2          EXE$V_CONCEALED : UNSIGNED (6); ! concealed device flag
317  0846  2
318  0847  2
```

MAKLOG
V04-000

G 10
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 8
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1 (3)

```
 319    0848  2 LINKAGE
 320    0849  2           ARGLST_IMGNAM   = JSB (REGISTER = 2;) :
 321    0850  2                             NOPRESERVE (0,1)
 322    0851  2                             NOTUSED (3,4,5,6,7,8,9,10,11),
 323    0852  2
 324    0853  2           EXE_CRE_GTABLE  = JSB (REGISTER = 11) :
 325    0854  2                             NOPRESERVE (0,1,2,3,4,5,8);
 326    0855  2
 327    0856  2 EXTERNAL ROUTINE
 328    0857  2           LOCK_IODB,                           ! lock the I/O data base
 329    0858  2           UNLOCK_IODB,                         ! unlock the I/O data base
 330    0859  2           NSA$EVENT_AUDIT : ADDRESSING_MODE (GENERAL),
 331    0860  2                                                ! security auditing routine
 332    0861  2           NSA$ARGLST_IMGNAM : ARGLST_IMGNAM ADDRESSING_MODE (GENERAL),
 333    0862  2                                                ! insert IMGNAM into ARGLST
 334    0863  2           EXE$CRE_GTABLE  : EXE_CRE_GTABLE ADDRESSING_MODE (GENERAL);
 335    0864  2                                                ! create group logical name table
 336    0865  2
 337    0866  2
 338    0867  2 ! First build the volume logical name table entry.
 339    0868  2 ! Use logical name from command unless:
 340    0869  2 !     - There is no logical name
 341    0870  2 !     - It is a disk volume set
 342    0871  2 !     - More than one device is being mounted, and they are not magtapes.
 343    0872  2 !
 344    0873  2 ! Get the logical name; either from the command or from the volume label.
 345    0874  2 !
 346    0875  2
 347    0876  2 ! Copy the user-specified logical name to local storage.
 348    0877  2 !
 349    0878  2
 350    0879  2 CH$MOVE (.LOG_NAME[0], .LOG_NAME[1], LOG_BUFFER);
 351    0880  2 NAME_DESC [DSC$W_LENGTH] = .LOG_NAME [0];
 352    0881  2 NAME_DESC [DSC$B_DTYPE] = 0;
 353    0882  2 NAME_DESC [DSC$B_CLASS] = 0;
 354    0883  2 NAME_DESC [DSC$A_POINTER] = LOG_BUFFER;
 355    0884  2
 356    0885  2 !
 357    0886  2 ! Calculate the access mode for the logical name creation
 358    0887  2 !
 359    0888  2
 360    0889  3 ACMODE = MIN ((IF .MOUNT_OPTIONS[OPT_SYSTEM]
 361    0890  2                  THEN PSL$C_EXEC
 362    0891  2                  ELSE PSL$C_SUPER), .CALLERS_ACMOD);
 363    0892  2
 364    0893  2 IF NOT .MOUNT_OPTIONS[OPT_LOG_NAME]
 365    0894  2 OR .SMTL_ENTRY NEQ 0
 366    0895  3 OR (.DEVICE_COUNT NEQ 1 AND (NOT .DEVICE_CHAR[DEV$V_SQD]))
 367    0896  3 THEN
 368    0897  3     BEGIN
 369    0898  3     IF .DEVICE_CHAR[DEV$V_SQD]
 370    0899  3     THEN P = TAPE_PREFIX
 371    0900  3     ELSE P = DISK_PREFIX;
 372    0901  3
 373    0902  3     C = LABEL_LENGTH (VCB$S_VOLNAME, VCB[VCB$T_VOLNAME]);
 374    0903  3     NAME_DESC[DSC$W_LENGTH] = .C + 5;
 375    0904  3     NAME_DESC[DSC$A_POINTER] = LOG_BUFFER;
```

MAKLOG
V04-000

H 10
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page   9
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (3)

```
376   0905   3          CH$COPY (5, .P, .C, VCB[VCB$T_VOLNAME], 0, .C+5, LOG_BUFFER);
377   0906              END;
378   0907
379   0908          ! Now create logical name.  The physical device string is the equivalence
380   0909          ! string.  If a tape mount, use the physical name of the first volume,
381   0910          ! otherwise use the physical name of the current volume.
382   0911
383   0912
384   0913          INDEX = .DEVICE_INDEX;
385   0914          IF .BBLOCK [UCB[UCB$L_DEVCHAR], DEV$V_SQD]
386   0915          THEN
387   0916              INDEX = 0;
388   0917
389   0918          ! Store the location of the LNM block in the MTL
390   0919
391   0920          ITEM_LIST [ 0 ] = ( LNM$_LNMB_ADDR^16 OR 4 );
392   0921          ITEM_LIST [ 1 ] = MTL_ENTRY[MTL$L_LOGNAME]; ! CAUTION USED BY ITEM_LIST [ 7 ]
393   0922          ITEM_LIST [ 2 ] = 0;
394   0923
395   0924          ! Store the location of the MTL in the LNM BLOCK.
396   0925          ! This causes the logical name deletion logic to clear the MTL's logical name
397   0926          ! pointer if the logical name is deleted, just as it does when a mailbox
398   0927          ! logical name is deleted.
399   0928          !
400   0929          ITEM_LIST [ 3 ] = ( LNM$_INDEX^16 or 4 );
401   0930          ITEM_LIST [ 4 ] = UPLIT ( LNMX$C_BACKPTR );
402   0931          ITEM_LIST [ 5 ] = 0;
403   0932          ITEM_LIST [ 6 ] = ( LNM$_STRING^16 or 4 );
404   0933          ITEM_LIST [ 7 ] = ITEM_LIST [ 1 ];
405   0934          ITEM_LIST [ 8 ] = 0;
406   0935
407   0936          ! Define equivalence string
408   0937          !
409   0938          ITEM_LIST [  9 ] = ( LNM$_INDEX^16 or 4 );
410   0939          ITEM_LIST [ 10 ] = UPLIT ( 0 );
411   0940          ITEM_LIST [ 11 ] = 0;
412   0941
413   0942          ITEM_LIST [ 12 ] = ( LNM$_ATTRIBUTES^16 or 4 );
414   0943          ITEM_LIST [ 13 ] = ( IF .EXE$GL_FLAGS[EXE$V_CONCEALED]
415   0944                                  THEN UPLIT ( LNM$M_TERMINAL OR LNM$M_CONCEALED )
416   0945                                  ELSE UPLIT ( LNM$M_TERMINAL ) );
417   0946          ITEM_LIST [ 14 ] = 0;
418   0947
419   0948          ! Use GETDVI to obtain the most universal device name for this physical
420   0949          ! device, FULLDEVNAM, and pass that to CRELNM as the equivalence name
421   0950          ! string.
422   0951          !
423   0952          PHYSNAM_DESC [ DSC$W_LENGTH ] = .PHYS_NAME [ .INDEX*2 ] - 1;
424   0953          PHYSNAM_DESC [ DSC$A_POINTER ] = .PHYS_NAME [ .INDEX*2 + 1 ] + 1;
425   0954          PHYSNAM_DESC [ DSC$B_DTYPE ] = 0;
426   0955          PHYSNAM_DESC [ DSC$B_CLASS ] = 0;
427   0956
428   0957          DVI_ITEM [ 0 ] = ( DVI$_FULLDEVNAM^16 or ( PHYS_LENGTH + 2 ) );
429   0958          DVI_ITEM [ 1 ] = FULLNAM;
430   0959          DVI_ITEM [ 2 ] = ITEM_LIST [ 15 ];
431   0960          DVI_ITEM [ 3 ] = 0;
432   0961          ITEM_LIST [ 15 ] = 0;
```

MAKLOG
V04-000

I 10
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 10
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (3)

```
433      0962  2
434   P  0963  2   $GETDVIW (
435   P  0964  2           devnam = PHYSNAM_DESC,
436      0965  2           itmlst = DVI_ITEM        );
437      0966  2
438      0967  2   IF .FULLNAM [ 0 ] eql %C'_'
439      0968  2        THEN BEGIN
440      0969  2            ITEM_LIST [ 15 ] = ( LNM$_STRING^16 or ( .ITEM_LIST [ 15 ] - 1 ) );
441      0970  2            ITEM_LIST [ 16 ] = FULLNAM + 1;
442      0971  2            END
443      0972  2        ELSE BEGIN
444      0973  2            ITEM_LIST [ 15 ] = ( LNM$_STRING^16 or .ITEM_LIST [ 15 ] );
445      0974  2            ITEM_LIST [ 16 ] = FULLNAM;
446      0975  2            END;
447      0976  2   ITEM_LIST [ 17 ] = 0;
448      0977  2
449      0978  2   ! End item list
450      0979  2
451      0980  2   ITEM_LIST [ 18 ] = 0;
452      0981  2
453      0982  2   !
454      0983  2   ! If the volume is to be mounted /group, then we have to create the group logical
455      0984  2   ! name in the group of the current process.  To avoid the situation that the group
456      0985  2   ! table does not exist, we call the EXE$CRE_GTABLE routine, which creates the group
457      0986  2   ! table if it doesn't already exist.
458      0987  2   !
459      0988  2   IF .MOUNT_OPTIONS [OPT_GROUP]
460      0989  2   THEN
461      0990  3       BEGIN
462      0991  3
463   P  0992  3       $FAO ( %ASCID 'LNM$GROUP_!OW',                ! Format LNM$GROUP_xxxxxx
464   P  0993  3              GROUP_TABLE,
465   P  0994  3              GROUP_TABLE,
466      0995  3              .(SCH$GL_CURPCB [PCB$L_UIC]) <16,16>); ! Convert our group number to octal
467      0996  3
468   P  0997  3       $FAO ( %ASCID '!OW',                          ! Format octal in ASCII
469   P  0998  3              ASC_GROUP_DESC,
470   P  0999  3              ASC_GROUP_DESC,
471      1000  3              .(SCH$GL_CURPCB [PCB$L_UIC]) <16,16>); ! Convert our group number to octal
472      1001  3
473      1002  3       EXE$CRE_GTABLE (ASC_GROUP);                   ! Create the LNM$GROUP_xxxxxx table
474      1003  3
475      1004  3       END;                                         ! exists
476      1005  3
477      1006  2
478   P  1007  2   $CRELNM
479   P  1008  2           ( ACMODE = ACMODE,
480   P  1009  2             TABNAM = (IF  .MOUNT_OPTIONS [ OPT_SYSTEM ]
481   P  1010  2                          THEN SYSTEM_TABLE
482   P  1011  2                          ELSE
483   P  1012  2                              IF  .MOUNT_OPTIONS [ OPT_GRCUP ]
484   P  1013  2                                  THEN GROUP_TABLE
485   P  1014  2                                  ELSE JOB_TABLE
486   P  1015  2                          ),
487   P  1016  2             LOGNAM = NAME_DESC,
488      1017  2             ITMLST = ITEM_LIST );
489      1018  2
```

```
 490   1019   2 ! Link the MTL entry into the list
 491   1020   2
 492   1021   2 MTL_ENTRY[MTL$L_UCB] = .UCB;
 493   1022   2 LOCK_IODB ();                                  ! lock the mount list
 494   1023   2
 495   1024   2 IF .MOUNT_OPTIONS[OPT_GROUP] OR .MOUNT_OPTIONS[OPT_SYSTEM]
 496   1025   2 THEN MOUNT_LIST = IOC$GQ_MOUNTLST[1]
 497   1026   2 ELSE
 498   1027   2     BEGIN
 499   1028   3     JIB = .SCH$GL_CURPCB[PCB$L_JIB];
 500   1029   3     MOUNT_LIST = JIB[JIB$L_MTLBL];             ! get the tail of the mount list
 501   1030   3     END;
 502   1031   2 INSQUE (.MTL_ENTRY, ..MOUNT_LIST);
 503   1032   2
 504   1033   2 UNLOCK_IODB ();                                ! unlock the mount list
 505   1034   2
 506   1035   2 ! Now build the volume set logical name if we are mounting volume 1 of a
 507   1036   2 ! disk volume set.
 508   1037   2 !
 509   1038   2 IF .$MTL_ENTRY NEQ 0
 510   1039   2 THEN
 511   1040   2     BEGIN
 512   1041   3
 513   1042   3     ! Get the logical name; either from the command or from the volume label.
 514   1043   3     !
 515   1044   3
 516   1045   3     ! Copy the user-specified logical name to local storage.
 517   1046   3     !
 518   1047   3     CH$MOVE (.LOG_NAME[0], .LOG_NAME[1], LOG_BUFFER);
 519   1048   3     NAME_DESC [DSC$W_LENGTH] = .LOG_NAME [0];
 520   1049   3     NAME_DESC [DSC$B_DTYPE] = 0;
 521   1050   3     NAME_DESC [DSC$B_CLASS] = 0;
 522   1051   3     NAME_DESC [DSC$A_POINTER] = LOG_BUFFER;
 523   1052   3
 524   1053   3     IF NOT .MOUNT_OPTIONS[OPT_LOG_NAME]
 525   1054   3     THEN
 526   1055   4         BEGIN
 527   1056   4         IF .DEVICE_CHAR[DEV$V_SQD]
 528   1057   4         THEN P = TAPE_PREFIX
 529   1058   4         ELSE P = DISK_PREFIX;
 530   1059   4
 531   1060   4         RVT = .VCB[VCB$L_RVT];
 532   1061   4         C = LABEL_LENGTH (RVT$S_STRUCNAME, RVT[RVT$T_STRUCNAME]);
 533   1062   4         NAME_DESC[DSC$W_LENGTH] = .C + 5;
 534   1063   4         NAME_DESC[DSC$A_POINTER] = LOG_BUFFER;
 535   1064   4         CH$COPY (5, .P, .C, RVT[RVT$T_STRUCNAME], 0, .C+5, LOG_BUFFER);
 536   1065   3         END;
 537   1066   3
 538   1067   3     ! Now create logical name.  The physical device string is the equivalence
 539   1068   3     ! string.  If a tape mount, use the physical name of the first volume,
 540   1069   3     ! otherwise use the physical name of the current volume.
 541   1070   3     !
 542   1071   3     INDEX = .DEVICE_INDEX;
 543   1072   3     IF .BBLOCK [UCB[UCB$L_DEVCHAR], DEV$V_SQD]
 544   1073   3     THEN
 545   1074   3         INDEX = 0;
 546   1075   3
```

```
547     1076    3       ! Store the location of the LNM block in the MTL
548     1077    3       !
549     1078    3       ITEM_LIST [ 0 ] = ( LNMS_LNMB_ADDR^16 OR 4 );
550     1079    3       ITEM_LIST [ 1 ] = $MTL_ENTRY[MTLSL_LOGNAME];
551     1080    3       ITEM_LIST [ 2 ] = 0;
552     1081    3
553     1082    3       ! Store the location of the MTL in the LNM BLOCK.
554     1083    3       ! This causes the logical name deletion logic to clear the MTL's logical
555     1084    3       ! name pointer if the logical name is deleted, just as it does when a
556     1085    3       ! mailbox logical name is deleted.
557     1086    3       !
558     1087    3       ITEM_LIST [ 3 ] = ( LNMS_INDEX^16 or 4 );
559     1088    3       ITEM_LIST [ 4 ] = UPLIT ( LNMXSC_BACKPTR );
560     1089    3       ITEM_LIST [ 5 ] = 0;
561     1090    3       ITEM_LIST [ 6 ] = ( LNMS_STRING^16 or 4 );
562     1091    3       ITEM_LIST [ 7 ] = ITEM_LIST [ 1 ];
563     1092    3       ITEM_LIST [ 8 ] = 0;
564     1093    3
565     1094    3       ! Define equivalence string
566     1095    3       !
567     1096    3       ITEM_LIST [ 9 ] = ( LNMS_INDEX^16 or 4 );
568     1097    3       ITEM_LIST [ 10 ] = UPLIT ( 0 );
569     1098    3       ITEM_LIST [ 11 ] = 0;
570     1099    3
571     1100    3       ITEM_LIST [ 12 ] = ( LNMS_ATTRIBUTES^16 or 4 );
572     1101    4       ITEM_LIST [ 13 ] = ( IF .EXE$GL_FLAGS[EXE$V_CONCEALED]
573     1102    4                               THEN UPLIT ( LNMSM_TERMINAL OR LNMSM_CONCEALED )
574     1103    3                               ELSE UPLIT ( LNMSM_TERMINAL ) );
575     1104    3       ITEM_LIST [ 14 ] = 0;
576     1105    3
577     1106    3       ! Use GETDVI to obtain the most universal device name for this physical
578     1107    3       ! device, FULLDEVNAM, and pass that to CRELNM as the equivalence name
579     1108    3       ! string.
580     1109    3       !
581     1110    3       PHYSNAM_DESC [ DSC$W_LENGTH ] = .PHYS_NAME [ .INDEX+2 ] - 1;
582     1111    3       PHYSNAM_DESC [ DSC$A_POINTER ] = .PHYS_NAME [ .INDEX*2 + 1 ] + 1;
583     1112    3       PHYSNAM_DESC [ DSC$B_DTYPE ] = 0;
584     1113    3       PHYSNAM_DESC [ DSC$B_CLASS ] = 0;
585     1114    3
586     1115    3       DVI_ITEM [ 0 ] = ( DVI$_FULLDEVNAM^16 or ( PHYS_LENGTH + 2 ) );
587     1116    3       DVI_ITEM [ 1 ] = FULLNAM;
588     1117    3       DVI_ITEM [ 2 ] = ITEM_LIST [ 15 ];
589     1118    3       DVI_ITEM [ 3 ] = 0;
590     1119    3       ITEM_LIST [ 15 ] = 0;
591     1120    3
592     1121  P 3       $GETDVIW (
593     1122  P 3               devnam = PHYSNAM_DESC,
594     1123    3               itmlst = DVI_ITEM       );
595     1124    3
596     1125    3       IF .FULLNAM [ 0 ] eql %C'_'
597     1126    3           THEN BEGIN
598     1127    4               ITEM_LIST [ 15 ] = ( LNMS_STRING^16 or ( .ITEM_LIST [ 15 ] - 1 ) );
599     1128    4               ITEM_LIST [ 16 ] = FULLNAM + 1;
600     1129    4               END
601     1130    4           ELSE BEGIN
602     1131    4               ITEM_LIST [ 15 ] = ( LNMS_STRING^16 or .ITEM_LIST [ 15 ] );
603     1132    4               ITEM_LIST [ 16 ] = FULLNAM;
```

MAKLOG
V04-000

L 10
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 13
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1 (3)

```
 604            1133    3                    END;
 605            1134                          ITEM_LIST [ 17 ] = 0;
 606            1135
 607            1136                          ! End item list
 608            1137
 609            1138                          ITEM_LIST [ 18 ] = 0;
 610            1139
 611      P     1140                          $CRELNM
 612      P     1141                                  ( ACMODE = ACMODE,
 613      P     1142                                    TABNAM = (IF  .MOUNT_OPTIONS [ OPT_SYSTEM ]
 614      P     1143                                              THEN SYSTEM_TABLE
 615      P     1144                                              ELSE
 616      P     1145                                                   IF  .MOUNT_OPTIONS [ OPT_GROUP ]
 617      P     1146                                                   THEN GROUP_TABLE
 618      P     1147                                                   ELSE JOB_TABLE
 619      P     1148                                             ),
 620      P     1149                                    LOGNAM = NAME_DESC,
 621            1150                                    ITMLST = ITEM_LIST );
 622            1151
 623            1152                          SMTL_ENTRY[MTL$L_UCB] = .UCB;
 624            1153                          SMTL_ENTRY[MTL$V_VOLSET] = 1;            ! identify as a volume set entry
 625            1154
 626            1155                          LOCK_IODB ();                           ! lock the mount list
 627            1156
 628            1157                          IF .MOUNT_OPTIONS[OPT_GROUP] OR .MOUNT_OPTIONS[OPT_SYSTEM]
 629            1158                          THEN MOUNT_LIST = IOC$GQ_MOUNTLST[1]
 630            1159                          ELSE
 631            1160    4                         BEGIN
 632            1161    4                         JIB = .SCH$GL_CURPCB[PCB$L_JIB];
 633            1162    4                         MOUNT_LIST = JIB[JIB$L_MTLBL];  ! get the tail of the mount list
 634            1163                              END;
 635            1164    3                     INSQUE (.SMTL_ENTRY, ..MOUNT_LIST);
 636            1165
 637            1166                          UNLOCK_IODB ();                         ! unlock the mount list
 638            1167
 639            1168    2                     END;
 640            1169
 641            1170
 642            1171            IF (.SCH$GL_CURPCB [PCB$V_SECAUDIT]
 643            1172    3       OR  .NSA$GR_ALARMVEC [NSA$V_EVT_MOUNT]
 644            1173    3       OR  .NSA$GR_JOURNVEC [NSA$V_EVT_MOUNT])
 645            1174    2       THEN
 646            1175                BEGIN
 647            1176
 648            1177    3           LOCAL
 649            1178                    ARGLIST : BBLOCK[NSA$K_ARG2_LENGTH],     ! security auditing argument list
 650            1179                    ORB     : REF BBLOCK,                    ! address of the ORB
 651            1180                    TEMP_PROT;                               ! temporary protection word
 652            1181
 653            1182    3           CH$FILL (0, NSA$K_ARG2_LENGTH, ARGLIST);    ! zero argument list
 654            1183    3           ORB = .UCB [UCB$L_ORB];                     ! get ORB address
 655            1184
 656            1185                    !
 657            1186                    ! Set up the security auditing argument list header
 658            1187                    !
 659            1188
 660            1189    3           ARGLIST [NSA$L_ARG_COUNT] = ( NSA$K_ARG2_LENGTH/4 ) - 4;
```

MAKLOG
V04-000

M 10
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 14
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (3)

```
 661   1190   3                                                          ! initialize length of argument list
 662   1191   3                                                          ! less vol-set pkt and arg count
 663   1192   3            ARGLIST [NSA$L_ARG_ID] = NSA$K_RECID_VOL_MOU;
 664   1193   3                                                          ! initialize record id as mount
 665   1194   3            IF .SCH$GL_CURPCB [PCB$V_SECAUDIT]             ! set up proper flags
 666   1195   3            THEN
 667   1196   3                ARGLIST [NSA$V_ARG_FLAG_MANDY] = 1;        ! mandatory auditing
 668   1197   3            IF .NSA$GR_ALARMVEC [NSA$V_EVT_MOUNT]
 669   1198   3            THEN
 670   1199   3                ARGLIST [NSA$V_ARG_FLAG_ALARM] = 1;        ! generate alarm for this record
 671   1200   3            IF .NSA$GR_JOURNVEC [NSA$V_EVT_MOUNT]
 672   1201   3            THEN
 673   1202   3                ARGLIST [NSA$V_ARG_FLAG_JOURN] = 1;        ! journal this record
 674   1203   3
 675   1204   3            ARGLIST [NSA$B_ARG_PKTNUM] = 7;                ! initialize number of items
 676   1205   3                                                          ! less vol-set pkt
 677   1206   3
 678   1207   3            !
 679   1208   3            ! Set up the security auditing argument list for mount
 680   1209   3            !
 681   1210   3
 682   1211   3            ARGLIST [NSA$L_ARG2_UIC_TM] = NSA$K_ARG_MECH_LONG^16 + NSA$K_PKTTYP_UIC;
 683   1212   3            ARGLIST [NSA$L_ARG2_UIC] = .ORB [ORB$L_OWNER];  ! set device owner UIC
 684   1213   3
 685   1214   3            ARGLIST [NSA$L_ARG2_VOLPRO_TM] = NSA$K_ARG_MECH_WORD^16 + NSA$K_PKTTYP_VOLPRO;
 686   1215   3            !
 687   1216   3            ! Get the volume protection
 688   1217   3            !
 689   1218   3            TEMP_PROT = 0;                                 ! clear temp location
 690   1219   3            IF .ORB [ORB$V_PROT_16]
 691   1220   3            THEN
 692   1221   3                TEMP_PROT = .ORB [ORB$W_PROT]              ! standard SOGW protection
 693   1222   3            ELSE
 694   1223   4                BEGIN                                     ! vector protection
 695   1224   4                TEMP_PROT <0,4>  = .(ORB [ORB$L_SYS_PROT])<0,4>;  ! system
 696   1225   4                TEMP_PROT <4,4>  = .(ORB [ORB$L_OWN_PROT])<0,4>;  ! owner
 697   1226   4                TEMP_PROT <8,4>  = .(ORB [ORB$L_GRP_PROT])<0,4>;  ! group
 698   1227   4                TEMP_PROT <12,4> = .(ORB [ORB$L_WOR_PROT])<0,4>;  ! world
 699   1228   3                END;
 700   1229   3            ARGLIST [NSA$L_ARG2_VOLPRO] = .TEMP_PROT;      ! set volume protection mask
 701   1230   3
 702   1231   3            ARGLIST [NSA$L_ARG2_MOUFLG_TM] = NSA$K_ARG_MECH_LONG^16 + NSA$K_PKTTYP_MOUFLG;
 703   1232   3            ARGLIST [NSA$L_ARG2_MOUFLG_] = .MOUNT_FLAGS;   ! set mount flags
 704   1233   3
 705   1234   3            NSA$ARGLST_IMGNAM (ARGLIST [NSA$L_ARG2_IMGNAM_TM]); ! set image name
 706   1235   3
 707   1236   3            ARGLIST [NSA$L_ARG2_DEVNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_DEVNAM;
 708   1237   3            IF .FULLNAM [0] EQL %C'_'
 709   1238   3            THEN
 710   1239   3                ITEM_LIST [15] = .ITEM_LIST [15] + 1;      ! include the '_' char
 711   1240   3            ARGLIST [NSA$L_ARG2_DEVNAM_SIZ] = .ITEM_LIST [15];  ! set size of full device name
 712   1241   3            ARGLIST [NSA$L_ARG2_DEVNAM_PTR] = FULLNAM;     ! set full device name buffer address
 713   1242   3
 714   1243   3            ARGLIST [NSA$L_ARG2_LOGNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_LOGNAM;
 715   1244   3            ARGLIST [NSA$L_ARG2_LOGNAM_SIZ] = .NAME_DESC [DSC$W_LENGTH];  ! set size of logical name
 716   1245   3            ARGLIST [NSA$L_ARG2_LOGNAM_PTR] = LOG_BUFFER;  ! set logical name buffer address
 717   1246
```

```
 718    1247  3          ARGLIST [NSA$L_ARG2_VOLNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_VOLNAM;
 719    1248  3          ARGLIST [NSA$L_ARG2_VOLNAM_SIZ] =
 720    1249  3              LABEL LENGTH (VCB$S_VOLNAME, VCB [VCB$T_VOLNAME]);   ! set size of volume name
 721    1250  3          ARGLIST [NSA$L_ARG2_VOLNAM_PTR] = VCB [VCB$T_VOLNAME];  ! set volume name buffer address
 722    1251  3
 723    1252  3
 724    1253  3          ! If the volume is a member of a volume set, then
 725    1254  3          !    a. increment argument count
 726    1255  3          !    b. increment number of packets
 727    1256  3          !    c. set up volume set descriptor
 728    1257  3          !
 729    1258  3
 730    1259  4          IF ( NOT .BBLOCK [UCB [UCB$L_DEVCHAR], DEV$V_FOR] )
 731    1260  4          AND    ( .VCB [VCB$W_RVN] NEQ 0 )
 732    1261  3          THEN
 733    1262  4              BEGIN
 734    1263  4              ARGLIST [NSA$L_ARG_COUNT] = .ARGLIST [NSA$L_ARG_COUNT] + 3; ! count vol-set pkt
 735    1264  4              ARGLIST [NSA$B_ARG_PKTNUM] = .ARGLIST [NSA$B_ARG_PKTNUM] + 1;
 736    1265  4              ARGLIST [NSA$L_ARG2_VOLSNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_VOLSNAM;
 737    1266  4              RVT = .VCB [VCB$L_RVT];
 738    1267  4              ARGLIST [NSA$L_ARG2_VOLSNAM_SIZ] =
 739    1268  4                  LABEL LENGTH (RVT$S_STRUCNAME, RVT [RVT$T_STRUCNAME]); ! set size of vol-set name
 740    1269  4              ARGLIST [NSA$L_ARG2_VOLSNAM_PTR] = RVT [RVT$T_STRUCNAME]; ! set vol-set name buffer address
 741    1270  3              END;
 742    1271  3
 743    1272  3          CALLG (ARGLIST, NSA$EVENT_AUDIT);   ! call event audit routine
 744    1273  3
 745    1274  2      END;                                   ! end of block defining ARGLIST
 746    1275  2
 747    1276  1 END;                                        ! end of routine ENTER_LOGNAME



                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                              24 45 50 41 54  00000 P.AAA:  .ASCII  \TAPE$\
                              24 4B 53 49 44  00005 P.AAB:  .ASCII  \DISK$\
                                              0000A          .BLKB   2
           00 00 4D 45 54 53 59 53 24 4D 4E 4C  0000C P.AAD:  .ASCII  \LNM$SYSTEM\<0><0>
                              010E000A  00018 P.AAC:  .LONG   17694730
                             00000000'  0001C          .ADDRESS P.AAD
                        00 42 4F 4A 24 4D 4E 4C  00020 P.AAF:  .ASCII  \LNM$JOB\<0>
                              010F0007  00028 P.AAE:  .LONG   17694727
                             00000000'  0002C          .ADDRESS P.AAF
  30 30 30 30 30 5F 50 55 4F 52 47 24 4D 4E 4C  00030 P.AAG:  .ASCII  \LNM$GROUP_000000\
                                    30  0003F
                              FFFFFF81  00040 P.AAH:  .LONG   -127
                              00000000  00044 P.AAI:  .LONG   0
                              00000300  00048 P.AAJ:  .LONG   768
                              00000200  0004C P.AAK:  .LONG   512
  00 00 57 4F 21 5F 50 55 4F 52 47 24 4D 4E 4C  00050 P.AAM:  .ASCII  \LNM$GROUP_!0W\<0><0><0>
                                    00  0005F
                              010E000D  00060 P.AAL:  .LONG   17694733
                             00000000'  00064          .ADDRESS P.AAM
                           00 57 4F 21  00068 P.AAO:  .ASCII  \!0W\<0>
                              010E0003  0006C P.AAN:  .LONG   17694723
                             00000000'  00070          .ADDRESS P.AAO
```

```
                    FFFFFF81  00074 P.AAP:  .LONG   -127
                    00000000  00078 P.AAQ:  .LONG   0
                    00000300  0007C P.AAR:  .LONG   768
                    00000200  00080 P.AAS:  .LONG   512

                                   TAPE_PREFIX=        P.AAA
                                   DISK_PREFIX=        P.AAB
                                   SYSTEM_TABLE=       P.AAC
                                   JOB_TABLE=          P.AAE
                                   .EXTRN  MOUNT_FLAGS, CALLERS_ACMOD
                                   .EXTRN  DEVICE_CHAR, DEVICE_COUNT
                                   .EXTRN  LOG_NAME, DEVICE_INDEX
                                   .EXTRN  PHYS_NAME, SMTL_ENTRY
                                   .EXTRN  SCH$GL_CURPCB, IOC$GQ_MOUNTLST
                                   .EXTRN  EXE$GL_FLAGS, NSA$GR_ALARMVEC
                                   .EXTRN  NSA$GR_JOURNVEC
                                   .EXTRN  EXE$V_CONCEALED
                                   .EXTRN  LOCK_IODB, UNLOCK_IODB
                                   .EXTRN  NSA$EVENT_AUDIT
                                   .EXTRN  NSA$ARGLST_IMGNAM
                                   .EXTRN  EXE$CRE_GTABLE, SYS$GETDVIW
                                   .EXTRN  SYS$FAO, SYS$CRELNM

                                   .PSECT  $CODE$,NOWRT,2

                    OFFC 00000     .ENTRY  ENTER LOGNAME, Save R2,R3,R4,R5,R6,R7,R8,-    ; 0734
                                           R9,R10,R11
                 5E   FDEC  CE 9E 00002     MOVAB   -532(SP), SP
0084  CE     0000' CF   10 28 00007         MOVC3   #16, P.AAG, TABLE_NAME               ; 0810
             7C AE        10 D0 0000F        MOVL    #16, GROUP_TABLE
             0080 CE 9E       00013          MOVAB   TABLE_NAME, GROUP_TABLE+4
             74 AE 30303030 8F D0 0001A      MOVL    #808464432, ASC_GROUP
             78 AE 30303030 8F D0 00022      MOVL    #808464432, ASC_GROUP+4
             6C AE            06 D0 0002A     MOVL    #6, ASC_GROUP_DESC
             70 AE    74 AE 9E 0002E          MOVAB   ASC_GROUP, ASC_GROUP_DESC+4
FEF8  CD  0000G DF  0000G CF 28 00033         MOVC3   LOG_NAME, @LOG_NAME+4, LOG_BUFFER  ; 0879
          F8 AD  0000G CF 3C 0003D            MOVZWL  LOG_NAME, NAME_DESC                ; 0880
          FC AD  FEF8 CD 9E 00043             MOVAB   LOG_BUFFER, NAME_DESC+4            ; 0883
          05  0000G CF E9 00049               BLBC    MOUNT_OPTIONS+1, 1$               ; 0889
          50    01 D0 0004E                   MOVL    #1, R0
          03 11 00051                         BRB     2$
          50    02 D0 00053 1$:               MOVL    #2, R0
          0000G CF 50 D1 00056 2$:            CMPL    R0, CALLERS_ACMOD                 ; 0891
          05 15 0005B                         BLEQ    3$
          50  0000G CF D0 0005D               MOVL    CALLERS_ACMOD, R0
          08 AE 50 D0 00062 3$:               MOVL    R0, ACMODE                        ; 0889
          13  0000G CF E1 00066               BBC     #5, MOUNT_OPTIONS+3, 4$           ; 0893
          0000G CF D5 0006C                   TSTL    SMTL_ENTRY                        ; 0894
          0D 12 00070                         BNEQ    5$
          01  0000G CF D1 00072               CMPL    DEVICE_COUNT, #1                  ; 0895
          58 13 00077                         BEQL    7$
          52  0000G CF E0 00079               BBS     #5, DEVICE_CHAR, 7$
          07  0000G CF E1 0007F 4$:           BBC     #5, DEVICE_CHAR, 5$               ; 0898
          6E  0000' CF 9E 00085               MOVAB   TAPE_PREFIX, P                    ; 0899
          05 11 0008A                         BRB     6$
          6E  0000' CF 9E 0008C 5$:           MOVAB   DISK_PREFIX, P                    ; 0900
          56    08 AC D0 00091 6$:            MOVL    VCB, R6                           ; 0902
```

```
                        14   A6 9F 00095            PUSHAB   20(R6)
                             0C DD 00098            PUSHL    #12
                 0000V CF    02 FB 0009A            CALLS    #2, LABEL_LENGTH
                    04 AE    50 D0 0009F            MOVL     R0, C                              0903
           50       04 AE    05 C1 000A3            ADDL3    #5, C, R0
                    F8 AD    50 B0 000A8            MOVW     R0, NAME_DESC                      0904
                    FC AD FEF8 CD 9E 000AC          MOVAB    LOG_BUFFER, NAME_DESC+4            0905
                             5A 50 D0 000B2         MOVL     R0, R10
                    57 FEF8 CD 9E 000B5             MOVAB    LOG_BUFFER, R7
  5A          00       00 BE 05 2C 000BA            MOVC5    #5, @P, #0, R10, (R7)
                             67    000C0
                             0E 18 000C1            BGEQ     7$
                    57       05 C0 000C3            ADDL2    #5, R7
                    5A       05 C2 000C6            SUBL2    #5, R10
  5A          00       14 A6 04 AE 2C 000C9         MOVC5    C, 20(R6), #0, R10, (R7)
                             67    000D0
                    56 0000G CF D0 000D1  7$:       MOVL     DEVICE_INDEX, INDEX                0913
                    57    04 AC D0 000D6            MOVL     UCB, R7                            0914
           02       38 A7 05 E1 000DA              BBC      #5, 56(R7), 8$
                    56 D4 000DF                     CLRL     INDEX                              0916
        00C4 CE  00C0 CE 00090004 8F D0 000E1  8$:  MOVL     #589828, ITEM_LIST                 0920
        00C4 CE  0000G CF 10 C1 000EA             ADDL3    #16, MTL_ENTRY, ITEM_LIST+4       0921
                    00C8 CE D4 000F2               CLRL     ITEM_LIST+8                        0922
           00CC CE 00010004 8F D0 000F6            MOVL     #65540, ITEM_LIST+12               0929
           00D0 CE  0000' CF 9E 000FF              MOVAB    P.AAH, ITEM_LIST+16               0930
                    00D4 CE D4 00106               CLRL     ITEM_LIST+20                      0931
           00D8 CE 00020004 8F D0 0010A            MOVL     #131076, ITEM_LIST+24             0932
           00DC CE  00C4 CE 9E 00113              MOVAB    ITEM_LIST+4, ITEM_LIST+28        0933
                    00E0 CE D4 0011A               CLRL     ITEM_LIST+32                      0934
           00E4 CE 00010004 8F D0 0011E            MOVL     #65540, ITEM_LIST+36             0938
           00E8 CE  0000' CF 9E 00127             MOVAB    P.AAI, ITEM_LIST+40              0939
                    00EC CE D4 0012E               CLRL     ITEM_LIST+44                     0940
           00F0 CE 00030004 8F D0 00132            MOVL     #196612, ITEM_LIST+48             0942
     07 00000000G 00 00G E1 0013B                 BBC      S^EXE$V_CONCEALED, EXE$GL_FLAGS, 9$   0943
                    50 0000' CF 9E 00143           MOVAB    P.AAJ, R0                         0944
                    05 11 00148                     BRB      10$
                    50 0000' CF 9E 0014A  9$:       MOVAB    P.AAK, R0                        0945
           00F4 CE 50 D0 0014F  10$:               MOVL     R0, ITEM_LIST+52                 0943
                    00F8 CE D4 00154               CLRL     ITEM_LIST+56                      0946
           50       56 01 78 00158               ASHL     #1, INDEX, R0                    0952
              0000GCF40 DF 0015C                   PUSHAL   PHYS_NAME[R0]
        00B8 CE       9E 01 A3 00161               SUBW3    #1, @(SP)+, PHYSNAM_DESC
        00BC CE 0000GCF40 01 C1 00167             ADDL3    #1, PHYS_NAME+4[R0], PHYSNAM_DESC+4   0953
                    00BA CE B4 00170               CLRW     PHYSNAM_DESC+2                   0954
           0094 CE 00E80011 8F D0 00174            MOVL     #15204369, DVI_ITEM              0957
           0098 CE  00A4 CE 9E 0017D              MOVAB    FULLNAM, DVI_ITEM+4             0958
           009C CE  00FC CE 9E 00184              MOVAB    ITEM_LIST+60, DVI_ITEM+8        0959
                    00A0 CE D4 0018B               CLRL     DVI_ITEM+12                      0960
                    00FC CE D4 0018F               CLRL     ITEM_LIST+60                     0961
                    7E 7C 00193                     CLRQ     -(SP)                             0965
                    7E 7C 00195                     CLRQ     -(SP)
                    00A4 CE 9F 00197               PUSHAB   DVI_ITEM
                    00CC CE 9F 0019B               PUSHAB   PHYSNAM_DESC
                    7E 7C 0019F                     CLRQ     -(SP)
        00000000G 00 08 FB 001A1                   CALLS    #8, SYS$GETDVIW
                    5F 8F 00A4 CE 91 001A8          CMPB     FULLNAM, #95                     0967
                    19 12 001AE                     BNEQ     11$
```

MAKLOG
V04-000

D 11
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 18
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (3)

```
            50      00FC  CE          01  C3 001B0          SUBL3    #1, ITEM_LIST+60, R0                        0969
    00FC  CE        0100  CE    00020000  BF  C9 001B6          BISL3    #131072, R0, ITEM_LIST+60                   0970
                    0100  CE          00A5  CE  9E 001C0          MOVAB    FULLNAM+1, ITEM_LIST+64                     0970
                                      0C  11 001C7          BRB      12$                                        0967
            00FE  CE                  02  88 001C9  11$:     BISB2    #2, ITEM_LIST+60                            0973
            0100  CE          00A4  CE  9E 001CE          MOVAB    FULLNAM, ITEM_LIST+64                       0974
            0104  CE          7C 001D5  12$:     CLRQ     ITEM_LIST+68                                0976
                    0000G  CF          95 001D9          TSTB     MOUNT_OPTIONS                              0988
                                      46  18 001DD          BGEQ     13$
            50 00000000G      00  D0 001DF          MOVL     SCH$GL_CURPCB, R0                          0995
                    7E        00BE  C0  3C 001E6          MOVZWL   190(R0), -(SP)
                              0080  CE  9F 001EB          PUSHAB   GROUP_TABLE
                              0084  CE  9F 001EF          PUSHAB   GROUP_TABLE
                              0000'  CF  9F 001F3          PUSHAB   P.AAL
    00000000G  00            04  FB 001F7          CALLS    #4, SYS$FAO
            50 00000000G      00  D0 001FE          MOVL     SCH$GL_CURPCB, R0                          1000
                    7E        00BE  C0  3C 00205          MOVZWL   190(R0), -(SP)
                      70      AE  9F 0020A          PUSHAB   ASC_GROUP_DESC
                      74      AE  9F 0020D          PUSHAB   ASC_GROUP_DESC
                              0000'  CF  9F 00210          PUSHAB   P.AAN
    00000000G  00            04  FB 00214          CALLS    #4, SYS$FAO
                      58      AE  9E 0021B          MOVAB    ASC_GROUP, R11                             1002
            00000000G  00    16 0021F          JSB      EXE$CRE_GTABLE
                              00C0  CE  9F 00225  13$:     PUSHAB   ITEM_LIST                                  1017
                      0C      AE  9F 00229          PUSHAB   ACMODE
                      F8      AD  9F 0022C          PUSHAB   NAME_DESC
            07        0000G  CF  E9 0022F          BLBC     MOUNT_OPTIONS+1, 14$
            50        0000'  CF  9E 00234          MOVAB    SYSTEM_TABLE, R0
                      12      11 00239          BRB      16$
                    0000G  CF          95 0023B  14$:     TSTB     MOUNT_OPTIONS
                      07      18 0023F          BGEQ     15$
            50        0088  CE  9E 00241          MOVAB    GROUP_TABLE, R0
                      05      11 00246          BRB      16$
            50        0000'  CF  9E 00248  15$:     MOVAB    JOB_TABLE, R0
                      50      DD 0024D  16$:     PUSHL    R0
                      7E      D4 0024F          CLRL     -(SP)
    00000000G  00            05  FB 00251          CALLS    #5, SYS$CRELNM
                    0C        0000G  CF  D0 00258          MOVL     MTL_ENTRY, R0                              1021
                    0C  A0            0000G  57  D0 0025D          MOVL     R7, 12(R0)
                    0000G  CF          00  FB 00261          CALLS    #0, LOCK_IODB                              1022
                              0000G  CF  95 00266          TSTB     MOUNT_OPTIONS                              1024
                      05      19 0026A          BLSS     17$
            09        0000G  CF  E9 0026C          BLBC     MOUNT_OPTIONS+1, 18$
            58 00000000G      00  9E 00271  17$:     MOVAB    IOC$GD_MOUNTLST+4, MOUNT_LIST              1025
                      10      11 00278          BRB      19$
            50 00000000G      00  D0 0027A  18$:     MOVL     SCH$GL_CURPCB, R0                          1028
                    5B        0080  C0  D0 00281          MOVL     128(R0), JIB
                    58        04  AB  9E 00286          MOVAB    4(R11), MOUNT_LIST                         1029
            00  B8            0000G  DF  0E 0028A  19$:     INSQUE   @MTL_ENTRY, @0(MOUNT_LIST)                1031
                    0000G  CF          00  FB 00290          CALLS    #0, UNLOCK_IODB                            1033
                              0000G  CF  D5 00295          TSTL     @MTL_ENTRY                                 1038
                      03      12 00299          BNEQ     20$
                      01ED    31 0029B          BRW      35$
    FEF8  CD        0000G  DF        0000G  CF  28 0029E  20$:     MOVC3    LOG_NAME, @LOG_NAME+4, LOG_BUFFER          1047
                    F8  AD            0000G  CF  3C 002A8          MOVZWL   LOG_NAME, NAME_DESC                        1048
                    FC  AD            FEF8  CD  9E 002AE          MOVAB    LOG_BUFFER, NAME_DESC+4                    1051
            56        0000G  CF  05  E0 002B4          BBS      #5, MOUNT_OPTIONS+3, 23$                   1053
```

```
              07    0000G  CF          05 E1 002BA         BBC     #5, DEVICE_CHAR, 21$              1056
                     6E       0000'     CF 9E 002C0         MOVAB   TAPE_PREFIX, P                   1057
                                        05 11 002C5         BRB     22$
                     6E       0000'     CF 9E 002C7  21$:   MOVAB   DISK_PREFIX, P                   1058
                     50          08     AC D0 002CC  22$:   MOVL    VCB, R0                          1060
                     59          20     A0 D0 002D0         MOVL    32(R0), RVT
                     0C                 A9 9F 002D4         PUSHAB  12(RVT)                          1061
                                        0C DD 002D7         PUSHL   #12
              0000V  CF                 02 FB 002D9         CALLS   #2, LABEL_LENGTH
                     04    AE           50 D0 002DE         MOVL    R0, C
              50     04    AE           05 C1 002E3         ADDL3   #5, C, R0                        1062
                     F8    AD           50 B0 002E7         MOVW    R0, NAME_DESC
                     FC    AD  FEF8     CD 9E 002EB         MOVAB   LOG_BUFFER, NAME_DESC+4          1063
                     5A                 50 D0 002F1         MOVL    R0, R10                          1064
                     57    FEF8         CD 9E 002F4         MOVAB   LOG_BUFFER, R7
  5A          00     00    BE           05 2C 002F9         MOVC5   #5, -@P, #0, R10, (R7)
                     67                       002FF
                                        0E 18 00300         BGEQ    23$
                     57                 05 C0 00302         ADDL2   #5, R7
                     5A                 05 C2 00305         SUBL2   #5, R10
  5A          00     0C    A9    04     AE 2C 00308         MOVC5   C, 12(RVT), #0, R10, (R7)
                     67                       0030F
                     56       0000G     CF D0 00310  23$:   MOVL    DEVICE_INDEX, INDEX             1071
                     52          04     AC D0 00315         MOVL    UCB, R2                         1072
              02     38    A2           05 E1 00319         BBC     #5, 56(R2), 24$
                     56                 56 D4 0031E         CLRL    INDEX                           1074
  00C4  CE    00C0  CE 00090004         8F D0 00320  24$:   MOVL    #589828, ITEM_LIST             1078
  00C4  CE    0000G  CF                 10 C1 00329         ADDL3   #16, SMTL_ENTRY, ITEM_LIST+4   1079
                            00C8         CE D4 00331         CLRL    ITEM_LIST+8                    1080
              00CC  CE 00010004         8F D0 00335         MOVL    #65540, ITEM_LIST+12           1087
              00D0  CE       0000'      CF 9E 0033E         MOVAB   P.AAP, ITEM_LIST+16           1088
                            00D4         CE D4 00345         CLRL    ITEM_LIST+20                  1089
              00D8  CE 00020004         8F D0 00349         MOVL    #131076, ITEM_LIST+24         1090
              00DC  CE       00C4       CE 9E 00352         MOVAB   ITEM_LIST+4, ITEM_LIST+28    1091
                            00E0         CE D4 00359         CLRL    ITEM_LIST+32                 1092
              00E4  CE 00010004         8F D0 0035D         MOVL    #65540, ITEM_LIST+36         1096
              00E8  CE       0000'      CF 9E 00366         MOVAB   P.AAQ, ITEM_LIST+40          1097
                            00EC         CE D4 0036D         CLRL    ITEM_LIST+44                 1098
              00F0  CE 00030004         8F D0 00371         MOVL    #196612, ITEM_LIST+48        1100
       07 00000000G    00            00G E1 0037A         BBC     S^EXE$V_CONCEALED, EXE$GL_FLAGS, 25$   1101
                     50       0000'     CF 9E 00382         MOVAB   P.AAR, R0                    1102
                                        05 11 00387         BRB     26$
                     50       0000'     CF 9E 00389  25$:   MOVAB   P.AAS, R0                    1103
              00F4  CE          50     D0 0038E  26$:   MOVL    R0, ITEM_LIST+52             1101
                            00F8         CE D4 00393         CLRL    ITEM_LIST+56                1104
                     56                 02 C4 00397         MULL2   #2, R6                      1110
                          0000GCF46     DF 0039A         PUSHAL  PHYS_NAME[R6]
  00B8  CE          9E                 01 A3 0039F         SUBW3   #1, @(SP)+, PHYSNAM_DESC
  00BC  CE    0000GCF46                 01 C1 003A5         ADDL3   #1, PHYS_NAME+4[R6], PHYSNAM_DESC+4   1111
                            00BA         CE B4 003AE         CLRW    PHYSNAM_DESC+2              1112
              0094  CE 00E80011         8F D0 003B2         MOVL    #15204369, DVI_ITEM         1115
              0098  CE       00A4       CE 9E 003BB         MOVAB   FULLNAM, DVI_ITEM+4         1116
              009C  CE       00FC       CE 9E 003C2         MOVAB   ITEM_LIST+60, DVI_ITEM+8   1117
                            00A0         CE D4 003C9         CLRL    DVI_ITEM+12                1118
                            00FC         CE D4 003CD         CLRL    ITEM_LIST+60               1119
                     7E                 7C 7C 003D1         CLRQ    -(SP)                      1123
                     7E                 7C 7C 003D3         CLRQ    -(SP)
```

```
                              00A4   CE 9F 003D5          PUSHAB   DVI_ITEM
                              00CC   CE 9F 003D9          PUSHAB   PHYSNAM_DESC
                                     7E 7C 003DD          CLRQ     -(SP)
                  00000000G 00       08 FB 003DF          CALLS    #8, SYS$GETDVIW
                        5F 8F 00A4   CE 91 003E6          CMPB     FULLNAM, #95            : 1125
                              19 12 003EC                 BNEQ     27$
             50 CE    00FC   CE      01 C3 003EE          SUBL3    #1, ITEM_LIST+60, R0    : 1127
      00FC CE         0100   CE 50 00020000 8F C9 003F4   BISL3    #131072, R0, ITEM_LIST+60
                             00A5   CE 9E 003FE          MOVAB    FULLNAM+1, ITEM_LIST+64  : 1128
                                    0C 11 00405          BRB      28$                      : 1125
                      00FE   CE      02 88 00407   27$:   BISB2    #2, ITEM_LIST+60         : 1131
                      0100   CE      CE 9E 0040C          MOVAB    FULLNAM, ITEM_LIST+64    : 1132
                             0104   CE 7C 00413   28$:   CLRQ     ITEM_LIST+68             : 1134
                             00C0   CE 9F 00417          PUSHAB   ITEM_LIST               : 1150
                                    0C AE 9F 0041B        PUSHAB   ACMODE
                                    F8 AD 9F 0041E        PUSHAB   NAME_DESC
                     07   0000G CF E9 00421              BLBC     MOUNT_OPTIONS+1, 29$
                     50   0000' CF 9E 00426              MOVAB    SYSTEM_TABLE, R0
                             12 11 0042B                 BRB      31$
                        0000G CF 95 0042D   29$:   TSTB     MOUNT_OPTIONS
                             07 18 00431                 BGEQ     30$
                     50   0088 CE 9E 00433              MOVAB    GROUP_TABLE, R0
                             05 11 00438                 BRB      31$
                     50   0000' CF 9E 0043A   30$:   MOVAB    JOB_TABLE, R0
                             50 DD 0043F   31$:   PUSHL    R0
                             7E D4 00441                 CLRL     -(SP)
                  00000000G 00 05 FB 00443              CALLS    #5, SYS$CRELNM
                        50 0000G CF D0 0044A            MOVL     SMTL_ENTRY, R0           : 1152
                     0C A0      52 D0 0044F            MOVL     R2, T2(R0)               : 1153
                     0B A0      01 88 00453            BISB2    #1, 11(R0)               : 1155
                  0000G CF      00 FB 00457            CALLS    #0, LOCK_IODB            : 1157
                        0000G CF 95 0045C              TSTB     MOUNT_OPTIONS
                             05 19 00460                 BLSS     32$
                     09   0000G CF E9 00462              BLBC     MOUNT_OPTIONS+1, 33$
                  58 00000000G 00 9E 00467   32$:   MOVAB    IOC$GQ_MOUNTLST+4, MOUNT_LIST : 1158
                             10 11 0046E                 BRB      34$
                  50 00000000G 00 D0 00470   33$:   MOVL     SCH$GL_CURPCB, R0          : 1161
                     5B   0080 C0 D0 00477              MOVL     128(R0), JIB
                  58      04 AB 9E 0047C              MOVAB    4(R11), MOUNT_LIST        : 1162
             00 B8   0000G DF 0E 00480   34$:   INSQUE   @SMTL_ENTRY, @0(MOUNT_LIST) : 1164
                  0000G CF      00 FB 00486            CALLS    #0, UNLOCK_IODB         : 1166
             56 00000000G 00 D0 0048B   35$:   MOVL     SCH$GL_CURPCB, R6              : 1171
          11   27 A6      03 E0 00492              BBS      #3, 39(R6), 36$             : 1172
          09 00000000G 00 01 E0 00497              BBS      #1, NSA$GR_ALARMVEC, 36$    : 1172
          01 00000000G 00 01 E0 0049F              BBS      #1, NSA$GR_JOURNVEC, 36$    : 1173
                             04 004A7                 RET
   0060 8F        00 6E 00 2C 004A8   36$:   MOVC5    #0, (SP), #0, #96, ARGLIST       : 1182
                                0C AE 004AF
                     53 04 AC D0 004B1              MOVL     UCB, R3                    : 1183
                     50 1C A3 D0 004B5              MOVL     28(R3), ORB
                     0C AE 14 D0 004B9              MOVL     #20, ARGLIST               : 1189
                  10 AE 00010008 8F D0 004BD        MOVL     #65544, ARGLIST+4          : 1192
          04   27 A6      03 E1 004C5              BBC      #3, 39(R6), 37$             : 1194
                14 AE      04 88 004CA            BISB2    #4, ARGLIST+8               : 1196
          04 00000000G 00 01 E1 004CE   37$:   BBC      #1, NSA$GR_ALARMVEC, 38$      : 1197
                14 AE      01 88 004D6            BISB2    #1, ARGLIST+8               : 1199
          04 00000000G 00 01 E1 004DA   38$:   BBC      #1, NSA$GR_JOURNVEC, 39$      : 1200
```

```
                    14    AE              02  88  004E2        BISB2   #2, ARGLIST+8                    1202
                    15    AE              07  90  004E6  39$:   MOVB    #7, ARGLIST+9                    1204
                    18    AE  0002000C    8F  D0  004EA        MOVL    #131084, ARGLIST+12              1211
                    1C    AE              60  D0  004F2        MOVL    (ORB), ARGLIST+16                1212
                    20    AE  0001000D    8F  D0  004F6        MOVL    #65549, ARGLIST+20               1214
                          51              D4  004FE        CLRL    TEMP_PROT                           1218
                    06          0B    A0  E9  00500        BLBC    11(ORB), 40$                        1219
                    51          18    A0  3C  00504        MOVZWL  24(ORB), TEMP_PROT                  1221
                          18          11  00508        BRB     41$
  51        04        00          18    A0  F0  0050A  40$:   INSV    24(ORB), #0, #4, TEMP_PROT         1224
  51        04        04          1C    A0  F0  00510        INSV    28(ORB), #4, #4, TEMP_PROT         1225
  51        04        08          20    A0  F0  00516        INSV    32(ORB), #8, #4, TEMP_PROT         1226
  51        04        0C          24    A0  F0  0051C        INSV    36(ORB), #12, #4, TEMP_PROT        1227
                    24    51              D0  00522  41$:   MOVL    TEMP_PROT, ARGLIST+24               1229
                    28    AE  0002000E    8F  D0  00526        MOVL    #131086, ARGLIST+28              1231
                    2C    AE  0000G       CF  D0  0052E        MOVL    MOUNT_FLAGS, ARGLIST+32          1232
                          52        30    AE  9E  00534        MOVAB   ARGLIST+36, R2                  1234
                    00000000G    00  16  00538        JSB     NSA$ARGLST_IMGNAM
                    3C    AE  00040005    8F  D0  0053E        MOVL    #262149, ARGLIST+48             1236
                    5F    8F          00A4    CE  91  00546        CMPB    FULLNAM, #95                1237
                          04          12  0054C        BNEQ    42$
                                00FC    CE  D6  0054E        INCL    ITEM_LIST+60                       1239
                    40    AE        00FC    CE  D0  00552  42$:   MOVL    ITEM_LIST+60, ARGLIST+52     1240
                    44    AE        00A4    CE  9E  00558        MOVAB   FULLNAM, ARGLIST+56            1241
                    48    AE  00040006    8F  D0  0055E        MOVL    #262150, ARGLIST+60             1243
                    4C    AE          F8    AD  3C  00566        MOVZWL  NAME_DESC, ARGLIST+64         1244
                    50    AE        FEF8    CD  9E  0056B        MOVAB   LOG_BUFFER, ARGLIST+68         1245
                    54    AE  00040007    8F  D0  00571        MOVL    #262151, ARGLIST+72             1247
                          52          08    AC  D0  00579        MOVL    VCB, R2                      1249
                                14    A2  9F  0057D        PUSHAB  20(R2)
                                0C    DD  00580        PUSHL   #12
                    0000V CF              02  FB  00582        CALLS   #2, LABEL_LENGTH
                    58    AE              50  D0  00587        MOVL    R0, ARGLIST+76
                    5C    AE          14    A2  9E  0058B        MOVAB   20(R2), ARGLIST+80           1250
                          2B          3B    A3  E8  00590        BLBS    59(R3), 43$                  1259
                                0E    A2  B5  00594        TSTW    14(R2)                             1260
                                26    13  00597        BEQL    43$
                    0C    AE          03    C0  00599        ADDL2   #3, ARGLIST                      1263
                                15    AE  96  0059D        INCB    ARGLIST+9                          1264
                    60    AE  00040008    8F  D0  005A0        MOVL    #262152, ARGLIST+84            1265
                          59          20    A2  D0  005A8        MOVL    32(R2), RVT                  1266
                                0C    A9  9F  005AC        PUSHAB  12(RVT)                            1268
                                0C    DD  005AF        PUSHL   #12
                    0000V CF              02  FB  005B1        CALLS   #2, LABEL_LENGTH
                    64    AE              50  D0  005B6        MOVL    R0, ARGLIST+88
                    68    AE          0C    A9  9E  005BA        MOVAB   12(RVT), ARGLIST+92          1269
  00000000G    00        0C    AE  FA  005BF  43$:   CALLG   ARGLIST, NSA$EVENT_AUDIT                 1272
                                04  005C7        RET                                                 1276
```

; Routine Size: 1480 bytes,   Routine Base: $CODE$ + 0015

MAKLOG
V04-000

H 11
16-Sep-1984 01:16:19   VAX-11 Bliss-32 V4.0-742   Page 22
14-Sep-1984 12:45:22   DISK$VMSMASTER:[MOUNT.SRC]MAKLOG.B32;1   (4)

```
  749    1277   1 ROUTINE LABEL_LENGTH (STR_LENGTH, STR_TEXT) =
  750    1278   1
  751    1279   1 !++
  752    1280   1 !
  753    1281   1 !   FUNCTIONAL DESCRIPTION:
  754    1282   1 !
  755    1283   1 !       This routine will return the length of a given string.
  756    1284   1 !       Trailing blanks at the end of the string are not counted
  757    1285   1 !       as part of the string.
  758    1286   1 !
  759    1287   1 !       NOTE THAT NO VOLUME MAY HAVE A VOLUME LABEL WITH TRAILING BLANKS.
  760    1288   1 !
  761    1289   1 !
  762    1290   1 !   CALLING SEQUENCE:
  763    1291   1 !       LABEL_LENGTH (ARG1, ARG2)
  764    1292   1 !
  765    1293   1 !   INPUT PARAMETERS:
  766    1294   1 !       ARG1: Input string length
  767    1295   1 !       ARG2: Input string address
  768    1296   1 !
  769    1297   1 !   IMPLICIT INPUTS:
  770    1298   1 !       NONE
  771    1299   1 !
  772    1300   1 !   OUTPUT PARAMETERS:
  773    1301   1 !       NONE
  774    1302   1 !
  775    1303   1 !   IMPLICIT OUTPUTS:
  776    1304   1 !       NONE
  777    1305   1 !
  778    1306   1 !   ROUTINE VALUE:
  779    1307   1 !       NONE
  780    1308   1 !
  781    1309   1 !   SIDE EFFECTS:
  782    1310   1 !       NONE
  783    1311   1 !
  784    1312   1 !--
  785    1313   1
  786    1314   2 BEGIN
  787    1315   2
  788    1316   2 MAP
  789    1317   2         STR_TEXT           : REF VECTOR [,BYTE];   ! Input string
  790    1318   2
  791    1319   2 LOCAL
  792    1320   2         PTR                : LONG;                 ! Pointer to current char.
  793    1321   2
  794    1322   2 ! Starting at the end of the string, decrement the string length
  795    1323   2 ! until a nonblank character is found, or the beginning of the string
  796    1324   2 ! is encountered.
  797    1325   2 !
  798    1326   2
  799    1327   2 PTR = .STR_LENGTH;
  800    1328   2 WHILE (.PTR GTR 0) AND (.STR_TEXT [.PTR-1] EQL %ASCII' ') DO
  801    1329   2     PTR = .PTR - 1;
  802    1330
  803    1331   3 RETURN (.PTR)
  804    1332   1 END;
```

```
                                  0000 00000 LABEL_LENGTH:
                                                     .WORD   Save nothing                  ; 1277
                        51      04  AC  D0 00002      MOVL    STR_LENGTH, PTR               ; 1327
                                    OF  15 00006 1$:  BLEQ    2$                            ; 1328
              50        51      08  AC  C1 00008      ADDL3   STR_TEXT, PTR, R0
                        20      FF  A0  91 0000D      CMPB    -1(R0), #32
                                    04  12 00011      BNEQ    2$
                                    51  D7 00013      DECL    PTR                           ; 1329
                                    EF  11 00015      BRB     1$
              50        51          51  D0 00017 2$:  MOVL    PTR, R0                       ; 1331
                                    04  0001A         RET                                   ; 1332
```

; Routine Size: 27 bytes,   Routine Base: $CODE$ + 05DD

```
; 805        1333  1
; 806        1334  1 END
; 807        1335  0 ELUDOM
```

PSECT SUMMARY

Name	Bytes	Attributes
$CODE$	1528	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
$PLIT$	132	NOVEC,NOWRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
_255DUA28:[SYSLIB]LIB.L32;1	18619	94	0	1000	00:01.9

COMMAND QUALIFIERS

;   BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:MAKLOG/OBJ=OBJ$:MAKLOG MSRC$:MAKLOG/UPDATE=(ENH$:MAKLOG)

; Size: 1528 code + 132 data bytes
; Run Time: 00:33.7

; Elapsed Time: 01:07.0
; Lines/CPU Min: 2376
; Lexemes/CPU-Min: 26826
; Memory Used: 345 pages
; Compilation Complete

CHKHM2
LIS

CHNUCB
LIS

GETUIC
LIS

ERASE
LIS

LEFTONE
LIS

MOUDK2
LIS

CHKHM1
LIS

CHKSM2
LIS

CLUSTRMNT
LIS

INIFC2
LIS

MOUDK1
LIS

MAKRUT
LIS

MAKLOG
LIS